**Minimum Jumps to Reach Home**

A certain bug's home is on the x-axis at position x. Help them get there from position 0.

The bug jumps according to the following rules:

* It can jump exactly a positions **forward** (to the right).
* It can jump exactly b positions **backward** (to the left).
* It cannot jump backward twice in a row.
* It cannot jump to any forbidden positions.

The bug may jump forward **beyond** its home, but it **cannot jump** to positions numbered with **negative** integers.

Given an array of integers forbidden, where forbidden[i] means that the bug cannot jump to the position forbidden[i], and integers a, b, and x, return *the minimum number of jumps needed for the bug to reach its home*. If there is no possible sequence of jumps that lands the bug on position x, return -1.

**Example 1:**

**Input:** forbidden = [14,4,18,1,15], a = 3, b = 15, x = 9

**Output:** 3

**Explanation:** 3 jumps forward (0 -> 3 -> 6 -> 9) will get the bug home.

**Example 2:**

**Input:** forbidden = [8,3,16,6,12,20], a = 15, b = 13, x = 11

**Output:** -1

**Example 3:**

**Input:** forbidden = [1,6,2,14,5,17,4], a = 16, b = 9, x = 7

**Output:** 2

**Explanation:** One jump forward (0 -> 16) then one jump backward (16 -> 7) will get the bug home.

**Constraints:**

* 1 <= forbidden.length <= 1000
* 1 <= a, b, forbidden[i] <= 2000
* 0 <= x <= 2000
* All the elements in forbidden are distinct.
* Position x is not forbidden.

[Discuss](https://leetcode.com/problems/minimum-jumps-to-reach-home/discuss)